UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/743,141	12/23/2003	Avinash Sodani	02207/17056	7479

23838 7590 08/09/2007

KENYON & KENYON LLP
1500 K STREET N.W.
SUITE 700
WASHINGTON, DC 20005

EXAMINER	
PETRANEK, JACOB ANDREW	
ART UNIT	**PAPER NUMBER**
2183	

MAIL DATE	DELIVERY MODE
08/09/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/743,141	SODANI ET AL.	
	Examiner	Art Unit	
	Jacob Petranek	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *18 July 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,2,4,5,7-15,18 and 20-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,2,4,5,7-15,18 and 20-24* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-2, 4-5, 7-15, 18, and 20-24 are pending.

2. The office acknowledges the following papers:

Claims and arguments filed on 7/18/2007.

Maintained Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. §103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-5, 7-12, 14-18, and 20-24 are rejected under 35 U.S.C. §103(a) as being unpatentable over Kadambi et al. (U.S. 6,934,830), in view of Choquette (U.S. 6,088,784), in view of Hennessy and Patterson ("Computer Architecture: A Quantitative Approach"), herein Hennessy.

5. As per claim 1:

Kadambi disclosed a processor comprising:

A register file (Kadambi: Figure 1 element 104, column 3 lines 6-12);

An execution unit (Kadambi: Figure 1 element 106, column 2 lines 66-67); and

A register file cache coupled to the register file and to the execution unit (Kadambi: Figure 1 element 102, column 3 lines 13-33)(A register file cache is a storage unit that stores instruction operands. The register pane is a storage unit smaller than the register file that stores instruction operands); and

Kadambi failed to teach a register file cache including a fill cache and a write-back cache; a write-back mechanism to move data from the write-back cache of the register file cache to the register file; and wherein said fill cache is to store a source operand from the register file if the source operand is not found in the fill cache and the write-back cache.

However, Choquette disclosed a write-back mechanism to move data from the first data storage structure to the register file (Choquette: Figure 2 element 104, column 3 lines 56-64)(The global bypass unit has the ability to select an instruction result either staying in the bypass unit or being written back to the register file. The combination results in the register pane of Kadambi having the ability to skip writing register results back to the register file and store them in the register pane for immediate or future use.).

The processor of Kadambi writes all execution results into both the register pane and the register file. While this is a simple policy of keeping the two memories synchronized, it's a very wasteful process in terms of power consumption. The processor of Choquette eliminates most of these writes by determining which execution results should be stored within the global bypass structure and which results should be written back to the register file. This process results in many fewer writes to the register file because the results in the global bypass will likely be used many times before needing to be written back. The advantage of saving power consumption would have motivated one of ordinary skill in the art at the time of the invention to implement the register pane of Kadambi with the ability to selectively write back instruction results. Thus, it would have been obvious to one of ordinary skill in the art at the time of the

invention to implement the register pane that can write back instruction results selectively for the advantage of reduced power consumption.

Kadambi and Choquette failed to teach a register file cache including a fill cache and a write-back cache; data moved from the write-back cache to the register file; and wherein said fill cache is to store a source operand from the register file if the source operand is not found in the fill cache and the write-back cache.

However, Hennessy disclosed a register file cache including a fill cache and a write-back cache (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. The first data block is the fill cache and the second data block is the write-back cache.);

Data moved from the write-back cache to the register file (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. The second data block moves data from the cache to a lower-memory. When combined with Choquette, the data is moved to the register file.); and

Wherein said fill cache is to store a source operand from the register file if the source operand is not found in the fill cache and the write-back cache (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. Both data blocks are searched for a hit. If a miss occurs, the data is fetched from lower memory, being the register file when combined with Kadambi and Choquette.).

The register pane and global bypass of Kadambi and Choquette respectively act as a data cache that stores a small number of operands from the register file in order to allow for quicker access to the data (Kadambi: Column 1 lines 57-67 continued to column 2 lines 1-5). Neither the register pane nor the global bypass show the details of how data is inputted and outputted, as well as how data is ejected when a miss occurs. This would have motivated one of ordinary skill in the art to look for additional details to find how a data cache is organized. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the details of the Alpha 21264 data cache into the register pane and global bypass combined unit.

6. As per claim 2:

Kadambi, Choquette, and Hennessy disclosed the processor of claim 1, wherein the write-back cache is to receive a result of an instruction executed by the execution unit (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. The second data block is capable of receiving data from the processor.).

7. As per claim 4:

Kadambi disclosed an apparatus comprising:

A first data storage structure to hold instruction operands (Kadambi: Figure 1 element 102, column 3 lines 13-33)(The register pane is a data storage structure that holds instruction operands.);

A second data storage structure to hold instruction operands, coupled to the first data storage structure (Kadambi: Figure 1 element 104, column 3 lines 6-12)(The

register file is a data storage structure that holds instruction operands.); and

A logic device coupled to the first data storage structure and to the second data storage structure, to execute instructions using operands read from either the first data structure (Kadambi: Figure 3 elements 308 and 314, column 4 lines 24-39)(Instructions are executed using operands from the register pane. Operands needed that aren't present in the register pane are forwarded from the register file to the register pane so that an instruction can reissue.); and

Kadambi failed to teach a first data storage structure including a fill portion and a write-back portion to hold instruction operands; wherein said fill portion is to store a first instruction operand from the second data storage structure if the first instruction operand is not found in the fill portion and the write-back portion of the first data storage structure; and a write-back mechanism to move data from the write-back portion of the first data storage structure to the second data storage structure.

However, Choquette disclosed a write-back mechanism to move data from the first data storage structure to the register file (Choquette: Figure 2 element 104, column 3 lines 56-64)(The global bypass unit has the ability to select an instruction result either staying in the bypass unit or being written back to the register file. The combination results in the register pane of Kadambi having the ability to skip writing register results back to the register file and store them in the register pane for immediate or future use.).

The processor of Kadambi writes all execution results into both the register pane and the register file. While this is a simple policy of keeping the two memories synchronized, it's a very wasteful process in terms of power consumption. The

processor of Choquette eliminates most of these writes by determining which execution results should be stored within the global bypass structure and which results should be written back to the register file. This process results in many fewer writes to the register file because the results in the global bypass will likely be used many times before needing to be written back. The advantage of saving power consumption would have motivated one of ordinary skill in the art at the time of the invention to implement the register pane of Kadambi with the ability to selectively write back instruction results. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the register pane that can write back instruction results selectively for the advantage of reduced power consumption.

Kadambi and Choquette failed to teach a first data storage structure including a fill portion and a write-back portion to hold instruction operands; wherein said fill portion is to store a first instruction operand from the second data storage structure if the first instruction operand is not found in the fill portion and the write-back portion of the first data storage structure; and data moved from the write-back cache of the first data storage structure to the second data storage structure.

However, Hennessy disclosed a first data storage structure including a fill portion and a write-back portion to hold instruction operands (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. The first data block is the fill cache and the second data block is the write-back cache.);

Data moved from the write-back cache of the first data storage structure to the

second data storage structure (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. The second data block moves data from the cache to a lower-memory. When combined with Choquette, the data is moved to the register file.); and

Wherein said fill portion is to store a first instruction operand from the second data storage structure if the first instruction operand is not found in the fill portion and the write-back portion of the first data storage structure (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. Both data blocks are searched for a hit. If a miss occurs, the data is fetched from lower memory, being the register file when combined with Kadambi and Choquette.).

The register pane and global bypass of Kadambi and Choquette respectively act as a data cache that stores a small number of operands from the register file in order to allow for quicker access to the data (Kadambi: Column 1 lines 57-67 continued to column 2 lines 1-5). Neither the register pane nor the global bypass show the details of how data is inputted and outputted, as well as how data is ejected when a miss occurs. This would have motivated one of ordinary skill in the art to look for additional details to find how a data cache is organized. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the details of the Alpha 21264 data cache into the register pane and global bypass combined unit.

8. As per claim 5:

Kadambi, Choquette, Hennessy disclosed the apparatus of claim 4, further comprising:

A data-management mechanism to move data corresponding to an operand from the second data storage structure to the logic device when the data is not present in the first data storage structure (Kadambi: Figure 3 element 308, column 4 lines 36-39).

9. As per claim 7:

Kadambi, Choquette, Hennessy disclosed the apparatus of claim 4, wherein the write-back mechanism moves the data based on a frequency of access to the data (Kadambi: Figure 3 element 310, column 4, lines 40-49)(The replaced entries are chosen based on a LRU scheme. The combination of the global bypass and register pane functionality results in data being moved back based on a LRU scheme if the register pane is full of operands.).

10. As per claim 8:

Kadambi, Choquette, Hennessy disclosed the apparatus of claim 4, wherein the write-back portion is to store write results of instructions executed by the logic device (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. The second data block also receives data from the processor.).

11. As per claim 9:

Kadambi, Choquette, Hennessy disclosed the apparatus of claim 5, wherein the data-management mechanism is to copy the data from the second data storage structure to the fill portion (Kadambi: Figure 3 element 308, column 4 lines 36-39)(The

register file writes operands to the register pane if an instruction needs an operand not currently present in the register pane.).

12. As per claim 10:

Kadambi, Choquette, Hennessy disclosed the apparatus of claim 4.

Kadambi, Choquette, Hennessy failed to teach wherein the first data storage structure is more ported than is the second data storage structure.

However, it would have been obvious to one of ordinary skill in the art that the register file could have fewer ports than the combination of the register pane with the elements of the global bypass unit. One of ordinary skill in the art would see that the combination likely will lead to fewer writes being needed to the register file and fewer reads should be inherent within the processor of Kadambi because a majority of the operand reads are done from the register pane to ensure better performance. The resulting fewer register writes and reads to/from the register file would lead to the ability to have fewer read and write ports on the register file compared to the register pane. One of ordinary skill in the art would have been motivated at the time of the invention to have fewer ports on the register file in order to reduce power consumption and cost from the register file. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the register file with fewer ports than the register pane because of the advantages from reduced power consumption and costs of the register file.

13. As per claim 11:

Kadambi, Choquette, Hennessy disclosed the apparatus of claim 4.

Kadambi, Choquette, Hennessy failed to teach an allocation mechanism to allocate a register in the first data structure to which to write an instruction result, wherein the allocate mechanism is to allocate the register such that the result will be written to the register only when all outstanding reads of contents of the register have completed.

However, it would have been obvious to one of ordinary skill in the art at the time of the invention that an allocated register in the register pane can't be overwritten before all instructions have read the operand. Failing to do this will lead to the program not working correctly when an instruction gets the overwritten value of a register operand within the register pane instead of the original contents.

14. As per claim 12:

Kadambi, Choquette, Hennessy disclosed the apparatus of claim 11, wherein the write-back mechanism is to cooperate with the allocation mechanism such that previous contents of the register will have been moved to the second data structure before the contents are overwritten by the result (Choquette: Figure 2 element 104, column 3 lines 56-64)(It's inherent that the contents of the combined register pane and global bypass would be moved back to the register file before being overwritten with new contents from one of the execution units. If this did not happen, then the program would eventually fail from having execution contents lost before they were written back to the register file.).

15. As per claim 14:

Kadambi, Choquette, Hennessy disclosed the apparatus of claim 4, wherein the first data storage structure includes shared tracks (Kadambi: Figure 1 element 102, column 3 lines 13-33)(The bus lines are shared to write and read operands to the register pane.).

16. As per claim 15:

Kadambi disclosed a method comprising:

Arranging a register file cache to communicate with an execution unit and a register file, (Kadambi: Figure 1 elements 102, 104, and 106, column 2 lines 66-67 and column 3 lines 6-33)(A register file cache is a storage unit that stores instruction operands. The register pane is a storage unit smaller than the register file that will store instruction operands);

Searching the register file cache for an instruction operand of an instruction to be executed by the execution unit (Kadambi: Figure 3 element 304, column 3 lines 64-67 continued to column 4 lines 1-3); and

If the operand is found in the register file cache, reading the operand from the register file cache (Kadambi: Figure 3 element 314, column 4 lines 24-35); and

Kadambi failed to teach said register file cache including a fill cache and a write-back cache; Searching the fill cache and the write-back cache of the register file cache; if the operand is found in one of the fill cache and the write-back cache of the register file cache, reading the operand from the register file cache; periodically writing data from the register file cache to the register file; and writing the operand from the register file to the fill cache if the operand is not found in one of the fill cache and the write-back

cache of the register file cache.

However, Choquette disclosed periodically writing data from the register file cache to the register file (Choquette: Figure 2 element 104, column 3 lines 56-64)(The global bypass unit has the ability to select an instruction result either staying in the bypass unit or being written back to the register file. The combination results in the register pane of Kadambi having the ability to skip writing register results back to the register file and store them in the register pane for immediate or future use.).

The processor of Kadambi writes all execution results into both the register pane and the register file. While this is a simple policy of keeping the two memories synchronized, it's a very wasteful process in terms of power consumption. The processor of Choquette eliminates most of these writes by determining which execution results should be stored within the global bypass structure and which results should be written back to the register file. This process results in many fewer writes to the register file because the results in the global bypass will likely be used many times before needing to be written back. The advantage of saving power consumption would have motivated one of ordinary skill in the art at the time of the invention to implement the register pane of Kadambi with the ability to selectively write back instruction results. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the register pane that can write back instruction results selectively for the advantage of reduced power consumption.

Kadambi and Choquette failed to teach said register file cache including a fill cache and a write-back cache; searching the fill cache and the write-back cache of the

register file cache; if the operand is found in one of the fill cache and the write-back cache of the register file cache, reading the operand from the register file cache; and writing the operand from the register file to the fill cache if the operand is not found in one of the fill cache and the write-back cache of the register file cache.

However, Hennessy disclosed said register file cache including a fill cache and a write-back cache (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. The first data block is the fill cache and the second data block is the write-back cache.);

Searching the fill cache and the write-back cache of the register file cache (Hennessy: Pages 403-406, figure 5.7)(Step 2 in figure 5.7 involves searching for an operand in the fill cache and write-back cache.);

If the operand is found in one of the fill cache and the write-back cache of the register file cache, reading the operand from the register file cache (Hennessy: Pages 403-406, figure 5.7)(Step 3 involves generating a hit within the data blocks. If a hit occurs, the data is sent to the execution unit requesting the data when combined with Kadambi and Choquette.); and

Writing the operand from the register file to the fill cache if the operand is not found in one of the fill cache and the write-back cache of the register file cache (Hennessy: Pages 403-406, figure 5.7)(The combination of Hennessy, Kadambi, and Choquette results in the register file cache being organized like the Alpha 21264 data cache. Both data blocks are searched for a hit. If a miss occurs, the data is fetched

from lower memory, being the register file when combined with Kadambi and Choquette.).

The register pane and global bypass of Kadambi and Choquette respectively act as a data cache that stores a small number of operands from the register file in order to allow for quicker access to the data (Kadambi: Column 1 lines 57-67 continued to column 2 lines 1-5). Neither the register pane nor the global bypass show the details of how data is inputted and outputted, as well as how data is ejected when a miss occurs. This would have motivated one of ordinary skill in the art to look for additional details to find how a data cache is organized. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the details of the Alpha 21264 data cache into the register pane and global bypass combined unit.

17. As per claim 18:

Kadambi, Choquette, and Hennessy disclosed the method of claim 15, further comprising:

Executing the instruction (Kadambi: Figure 3 element 316, column 4 lines 24-35); and

Writing a result of the instruction to the register file cache (Kadambi: Figure 3 element 318, column 4 lines 24-35).

18. As per claim 20:

Kadambi, Choquette, and Hennessy disclosed the method of claim 15, wherein the data are written based on a least-recently-used policy (Kadambi: Figure 3 element 310, column 4 lines 40-49)(The combination of the register pane and the global bypass

unit replaces entries based on a LRU scheme.).

19. As per claim 21:

Kadambi, Choquette, and Hennessy disclosed the method of claim 18.

Kadambi, Choquette, and Hennessy failed to teach allocating a register in the register file cache to which to write the instruction result, such that the result will be written to the register only when all outstanding reads of contents of the register have completed.

However, it would have been obvious to one of ordinary skill in the art at the time of the invention that an allocated register in the register pane can't be overwritten before all instructions have read the operand. Failing to do this will lead to the program not working correctly when an instruction gets the overwritten value of a register operand within the register pane instead of the original contents.

20. As per claim 22:

Kadambi, Choquette, and Hennessy disclosed the method of claim 18, further comprising:

Allocating a register in the register file cache to which to write the instruction result (Kadambi: Figure 3 element 318, column 4 lines 24-35);

Periodically writing data from the register file cache to the register file (Choquette: Figure 2 element 104, column 3 lines 56-64)(A register file cache is a storage unit that stores instruction operands. The combined global bypass unit and register pane stores instruction operands and has the ability to either write instruction results back to the register file or store the results in the global bypass unit.); and

Timing the allocating and the periodic writing such that previous contents of the register will have been moved to the register file before the contents are overwritten by the result (Choquette: Figure 2 element 104, column 3 lines 56-64)(It's inherent that the contents of the combined register pane and global bypass would be moved back to the register file before being overwritten with new contents from one of the execution units. If this did not happen, then the program would eventually fail from having execution contents lost before they were written back to the register file.).

21. As per claim 23:

Claim 23 essentially recites the same limitations of claim 1. Claim 23 additionally recites the following limitations:

A memory to hold instructions for execution (Kadambi: Figure 1 element 108, column 3 lines 2-5).

22. As per claim 24:

Claim 24 essentially recites the same limitations of claim 2. Therefore, claim 24 is rejected for the same reasons as claim 2.

23. Claim 13 is rejected under 35 U.S.C. §103(a) as being unpatentable over Kadambi et al. (U.S. 6,934,830), in view of Choquette (U.S. 6,088,784), in view of Hennessy and Patterson ("Computer Architecture: A Quantitative Approach"), herein Hennessy, further in view of Zaitzeva et al. (U.S. 5,781,924).

24. As per claim 13:

Kadambi, Choquette, and Hennessy disclosed the apparatus of claim 4.

Kadambi, Choquette, and Hennessy failed to teach wherein the write-back portion comprises a first section and a second section, each of the first and second sections being divided into a plurality of subsections, wherein a subsection of the first section and a subsection of the second section have an exclusive set of write paths thereto.

However, Zaitzeva disclosed the first data storage structure comprises a first section and a second section, each of the first and second sections being divided into a plurality of subsections, wherein a subsection of the first section and a subsection of the second section have an exclusive set of write paths thereto (Zaitzeva: Figure 1 element 110, column 2 lines 8-65)(Each port writes to a separate section that contains subsections.).

The processor of Kadambi contains a register pane that is a cache smaller in size of the register file (Kadambi: Column 3 lines 13-33). The register pane can be arranged in a set-associative manner (Kadambi: Column 4 lines 4-23). Kadambi failed to teach how the actual implementation of a set-associative cache works. However, Zaitzeva disclosed a set-associative cache that works on multiple ports. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to search to find out a set-associative cache functionally works, such as how the cache is implemented in Zaitzeva.

Response to Arguments

25. The arguments presented by Applicant in the response, received on 7/18/2007 are not considered persuasive:

26. Applicant argues that "Kadambi, Choquette, and Hennessy failed to teach the fill and write-back portions of the register file cache, wherein said fill cache is to store a source operand from the register file if the source operand is not found in the fill cache and the write-back cache and wherein the write-back cache is to receive a result of an instruction executed by the execution unit."

This argument is not found to be persuasive for the following reason. Applicant further states that "there is no reference to first and second data blocks in the Hennessy reference." The examiner used these terms to differentiate the two separate data blocks shown in figure 5.7 in Hennessy. These data blocks are both shown directly above the circle containing 4. Given the lack of individual numbering of all of the elements in figure 5.7, this seemed to be the most logical way of differentiating the elements in the rejection by the examiner. Each data block is capable of receiving data from a higher-level cache and sending operand data to the execution units when combined with Kadambi and Choquette. This is shown in figure 5.7, where the CPU data out line is input into each data block, which when combined with Kadambi and Choquette results in receiving an execution result. Additionally, there are lines being input to the data blocks that are output from the lower-level memory, which when combined with Kadambi and Choquette results in receiving a source operand from the register file.

The examiner notes that further clarification to the claims for the fill cache and write-back cache could overcome the current rejections. The applicant could further describe the write-back cache as not being able to receive operand data from the register file and the fill cache as not being able to receive destination operand data from the execution units. This would overcome Hennessy since the data blocks of Hennessy each perform both operations.

Conclusion

THIS ACTION IS MADE FINAL.

The following is text cited from 37 CFR 1.111(c): In amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jacob Petranek whose telephone number is 571-272-5988. The examiner can normally be reached on M-F 8:00-4:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Jacob Petranek
Examiner, Art Unit 2183

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100